International Journal of Science Engineering and Advance Technology, IJSEAT, Vol. 4, Issue 1 | ISSN 2321-6905 January -2016

International Journal of Science Engineering and Advance Technology

A Novel Approach for Consistent Self Organizing By Using Agile Methods

G.Lakshmi Mounica, T.Sai Durga
Student, Kakinada Institute of Engineering & Technology Womens, Korangi.
Associate Professor in Department of Computer science engineering,
Kakinada institute of Engineering and Technology for women.

Abstract: Agile processes are highly talented activities in Software Engineering. These processes has been implementing successfully to increase the customer satisfaction and to decrease the response time. The advantages of these processes are also presented in this paper. The link exchange method between customer and developer by using Agile process was presented. Software development teams and their clients will have wide range of communication and execute their tasks as a consistent self-organizing team is presented in this paper. These teams have been recognized in socio-technical systems,, agents of knowledge management, and as examples of complex-adaptive systems. Over the last decade, self-organizing teams plays a major role stage in software environment when they were considerd as a mark of Agile methods. These roles Mentor, Coordinator, mediator, Champion, Promoter, and Terminator are supposed to provide their guidance and adherence to Agile methods, effectively reaching customer expectations and coordinating customer specifications, securing and sustaining senior management support, and identifying and terminating the team members with inconsistent self-organizing ability of the team.

Keywords: Agile Process; Software Engineering; Self-Organizing; Development Roles.

I.INTRODUCTION : Agile development methods have been implimented successfully to increase the customer satisfaction and to decrease the response time. This concept has gained wide range of importance in software environment. Agile models encourage rapid growth and have certain drawbacks, such as poor documentation and bad quality. Fast growth promotes use of these methods in small scale projects. Key concepts of agile methods are reliable and highly moderative with a strong importance on stakeholder contribution. The agile development groups maintain that agileconcepts that are are more fitting to what is actually required in industrial software development. Agile methodologies and practices come out as a crack to exact outcomes and openly hold advanced rates of change in software requirements and customer expectations. Some well-known agile methodologies are Adaptive Software Development, Crystal, Dynamic Systems Development Method, Extreme Programming (XP), Feature Driven Development (FDD), Pragmatic Programming, and Scrum.

Software development depends on common activities such as distributing ,methodologies, partitioning the tasks and responsibilities, avoiding the rate of bugs and increasing the overall quality of the software product. there must rapid communication between developers, continuous monitoring towards work. Self-organizing teams are important part which should be enabled in Agile environment.. Self-organizing teams are combining of “developers who develop their targeting tasks, and who share activities among themselves based on needs and ,who include their presence in decision making”. these teams must have single goal, mutual communication, and the ability to manage and meet new challenges. The approach mainly provides self organizing teams and the concept” teams has been added to XP. These teams not only enable Agile practices, but also capture the agile protocols, which targets on customer specifications. Self-organizing team is one of the best protocol behind the Agile Manifesto and have been identified as one of the important factor of Agile projects.

II. LITERATURE REVIEW: Many researchers studied and focused on monitoring the evolution and progress of software projects in agile software processes and Extreme Programming (XP). Some of them are Rostislav Fojtik [1] presentedthe using agile methods software development for a concrete application which is designed for clients with particular bugs. It shows the abilities and disabilities of different methods, particularly Extreme Programming. Laurie Williams [2] presented the principles that motivate and connect the agile methodologies. Software teams select the intersected agile practices and develop their own moderate software development technologies rather than compare all the practices of a inbuilt agile model. Lucas Layman et al [3] described a case study performed at Sabre Airline

SolutionsTM provide the results of adopting Extreme Programming (XP) concept with a team that had characteristically plan driven risk factors. Jean-Guy Schneider and Lorraine Johnston [4] examined the targeting goals for noval software environment and considerable practices of extreme Programming. Cristiano Tolfo and Raul Sidnei Wazlawick [5] reported the adoption of extreme programming (XP) method requires a very conventional context in software development companies. Helen Sharp and Hugh Robinson [6] analyzed the structure and use of story cards and the reliable aspects of the model, complete specifications of teams, using distributed approach. Görel Hedin et al [7] described the point of how extreme programming fits into the highly moderated and advanced level of software engineering curriculum. Mohamma d Alshaye and Wei Li [8] delivered the relationship among three XP engineering activities: new design, refactoring, and error fix. Christina Hansson et al [9] described and then analyze their development practices of the software providers. Muhammad A.

Noor et al [10] presented approaches of adopting agile principles in product line planning. Petri Kettunen [11] explained the relationships between the key concepts of AM and agile software methods, and suggests potential new areas for software process improvement (SPI) in large scale organizations.
Rizwan Jameel Qureshi, M and Hussain, S, A [12] presented the extreme programming (XP) process model and proposed a novel adaptive process model. John McAvoy and Tom Butler [13] presented a qualitative concept of leunderstanding failures with the development of a new software methodology by a project team. Ivana Turnu et al [14] introduced a model of open source software development process. The incorporating agile process provide best output in the issue of logic. Witold Pedrycz [15] proposed logic protocols based upon the phenomenon of different valued and fuzzy logic. The realisation of such models gives chance to the networks that are easy to construct, collaborate and interpret.

III.PREPARE YOUR PAPER BEFORE STYLING:
The rules behind the Agile curiculum include fast, frequent, consistent, and wide production of working software; responding to differentiation of specifications encouraging better communication; and encouraging and well cooperating teams. Agile models are delivered as a result to the weaknesses of conventional software environment. Agile process increase the conventional software environment models by including changes through an iterative and incremental style of development, allowing every iteration to target on a small set of functionalities prioritized by the customer. Agile methods encourage continuous customer involvement and feedback, and allow the customer to prioritize the features they want developed first. Agile models are best implimented to small/medium-sized business systems or PC products. Problems of the Agile methods is it can be complex process to note the requirements of clients who are included in the task. The actors may be unfit for the intense involvement that specifiesthe agile methods. Order of the modifications can be difficult where there are multiple stakeholders.

Maintaining reliability needs huge activities. Contracts may be a problem as with different methods to moderative environment. The first priority is to satisfy the client by in time and best delivery of valuable software. Agile processes connect modifications for the customer`s competitive advantage.. Agile processes promote sustainable development. The clients, developers and users should be able to maintain a constant locality of using the delivered software. The conceptof the moderative agile software environment is to include the best principles with mutual, adaptive team work stratagies cooperated by efficient software engineering tools. The strength of the Agile process is shown in figure.

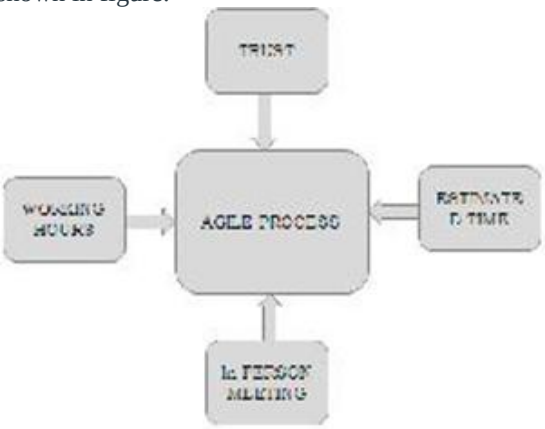

Fig:1. Strengths of the Agile Process
The strengths of the Agile process are trust, working hours and estimated time and in person meeting. Extreme Programming is derliver from Agile model but with different rules and protocols. When you work in a software development team, both of the above methodologies are so close.
Different people provide different opinion about the two methodologies.

A. Agile process Case Study Agile process can be useful to any software company, during the process of

development of the software project the changes in structure of the project is done with the agile process. Link exchange method was successfully implemented with Agile process method.

The link exchange method between customer and developer was shown in figure As the part of promoting their website one large company was using this method. A team of experts was searching the internet for relative websites to exchange links. The validity of the link was checked manually on a regular basis for the each link.

Effective software (i.e: Agile process) is required to develop the software services for the verification of the stock in links and check regularly. The customers are regularly updating their new requirements and ideas.

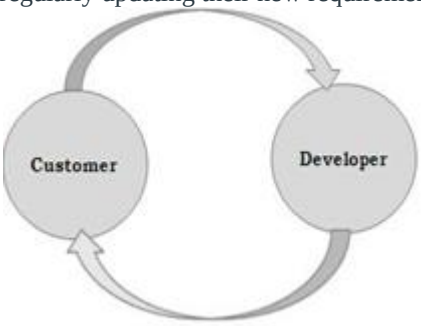

Fig:2 Link Exchange Method

For implementing the new ideas, Agile principles were observed while working at the project:

• Face-to-face communication was chosen to written records.
• The common environment to provide better daily cooperation for developer and customer.
• The active part of the customer in developing process.
• Mutual trust and respect to opinions between the developers and the customer.
• Merging all the old bases into one new base.
• Automatic check of the link validity.
• Providing different access levels.
• Optimizing work process.

focused specifically toward self-organization and are performed over and above the organizational roles. The self-organizing roles are informal and implicit because, they are not formally designated to the actors who operate them. The self-organizing tasks are transient because,they include in response to challenges faced by the Agile team and disappear or become dormant as the problems subside. The self organizing is spontaneous because, they are intuitively picked up by different actors of the team.

A. Getting the Team Confident

As the team moves through sprints or iterations, they become more eligible in understanding and practice of

IV.AGILE SOFTWARE DEVELOPME -NT PERSPECTIVE

An Agile software development perspective, self-organizing teams are considered the source of the best architecture, requirements, and design. Self-organization Manifesto,

. Sutherland, a cocreator of Scrum, explains that selforganizing teams consist of “members with diverse backgrounds” who are “given a free hand” by the top management. Schwaber, the other cocreator of Scrum, says that Agile methods “employ self-organizing teams” which are cross functional, not limited by their organizational job titles, training, or experience, rather the team “self-organizes based on its strengths and weaknesses to do the work at hand”. Schwaber suggests individuals on the team need to coordinate their individual self-organization with the rest of the team via daily synchronization meetings called daily Scrums.

V.SELF-ORGANIZING TEAM ROLES

This section describes the key results of the study the self-organizing roles that exist on Agile software development teams. Members of software development teams, both Agile and non-Agile, fulfill organizational roles on the team. For example, developers are responsible for development, testers are responsible for testing, business analysts are responsible for requirements analysis, etc.

In Agile teams, however, these organizational roles are not strictly adhered to, and members often function outside their boundaries when organizing themselves. Members of Agile teams play one or more of six informal, implicit, transient, and spontaneous roles in order to self-organize. These self-organizing Agile team roles—Mentor, Coordinator, Translator, Champion, Promoter, and Terminator—are the main actors involving in the process and the roles are to be

Agile concepts Demonstrations of working software to the customers and receiving feedback from them at the end of the sprint become important sources of positive reinforcement for the new team. The Mentor encourages the team to take the feedback in a constructive spirit and use it to moderate the practices. “When you get the team used to success, that’s where a change happens in them. You’ll encourage the models they haven’t done this before, they exit it. You need to show them...that they have achieved something, that they had a client presentation and the task that is completed...And final iteration...they gain the confidence...And after a few such validation cycles, then they start to get confident”.

> Encouraging iterative models or noval developers of the team, with no previous software development experience, find it easier to adopt Agile practices. "I find that there are perfectly capable developers are not bothered to change anymore. They have reached a certain level of perceived mastery and they're not at all driven to excel or to challenge themselves...And conversely, you have hungry people [fresh developers] that don't know any better just yet and you can show them a way to do better, and they do.

The eligible developers, with best experience of working with non-Agile software development methods, have a focus to convert to their old ways in the initial stages. "Actually it takes a wide effort for a team to become self-organizing, specially if people are coming from conventional software environment methods. It takes time, specially because I've worked with [company name] and even in [this company] you see people they come from traditional, they are into a habit of work which is very complicated to start with.". the major cncept of the Mentor activity is to enhance the cooperative goal of continued respective of the Agile process and methodologies. The below concept describes a process where the Mentor was initially go through the environment adopted by the team to be self-organizing.This tend to the huge mistake. In the absence of a Mentor, the team lost the importance of retrospectives.

CONCLUSION:

Agile concepts have been impimented successfully to increase the customer satisfaction and to decrease the response time. The strengths of the Agile methods are also presented in this paper. Self-organizing teams are the major part of agile development. The lack of research into self-organizing roles in software environment tend to is little diveation towards the Agile teams organize themselves in practice. Mentor: who guides and supports the team initially, encourage them to have enough stuff in Agile concepts,, ensures wide focus to Agile methods, and provide a way to the development of selforganizing practices in the team. Coordinator: who acts as a representative of the team to manage client requirements and coordinate customer collaboration with the team. It also presented a self-organizing teams from multiple perspectives and discussed the implications of our findings for practitioners. Going through these guidlines will help developers and their managers to execute their tasksself-organizing .

organizing team.

REFERENCES

[1] Rostislav Fojtik "Extreme Programming in development of specific software" Procedia Computer Science, Vol. 3, 2011, pp 1464-1468.

[2] Laurie Williams "Agile Software Development Methodologies and Practices" Advances in Computers, Vol. 80, 2010, pp 1-44.

[3] Lucas Layman., Laurie Williams and Lynn Cunningham "Motivations and measurements in an agile case study" Journal of Systems Architecture, Vol. 52, Issue 11, November 2006, pp 654-667. [4] Jean-Guy Schneider and Lorraine Johnston "eXtreme Programming helpful or harmful in educating undergraduates?" Journal of Systems and Software, Vol. 74, Issue 2, January 2005, pp 121-132.

[5] Cristiano Tolfo and Raul Sidnei Wazlawick "The influence of organizational culture on the adoption of extreme programming" Journal of Systems and Software, Vol. 81, Issue 11, November 2008, pp 1955-1967.

[6] Helen Sharp and Hugh Robinson "Collaboration and co-ordination in mature eXtreme programming teams" International Journal of Human-Computer Studies, Vol. 66, Issue 7, July 2008, pp 506-518.

[7] Görel Hedin., Lars Bendix and Boris Magnusson "Teaching extreme programming to large groups of students" Journal of Systems and Software, Vol. 74, Issue 2, January 2005, pp 133-146.

[8] Mohammad Alshaye and Wei Li "An empirical study of relationships among extreme programming engineering activities" Information and Software Technology, Vol. 48, Issue 11, November 2006, pp 1068-1072.

[9] Christina Hansson., Yvonne Dittricha., Björn Gustafsson and Stefan Zarnak "How agile are industrial software development practices?" Journal of Systems and Software, Vol. 79, Issue 9, September 2006, pp 1295-1311.

[10] Muhammad A. Noor., Rick, R., Paul Grünbacher "Agile product line planning: A collaborative approach and a case study" Journal of Systems and Software, Vol. 81, Issue 6, June 2008, pp 868-882.

[11] Petri Kettunen "Adopting key lessons from agile manufacturing to agile software product development—A comparative study" Technovation, Vol. 29, Issues 6-7, June 2009, pp 408-422.

[12] Rizwan Jameel Qureshi, M and Hussain, S, A "An adaptive software development process model" Advances in Engineering Software, Vol. 39, Issue 8, August 2008, pp 654-658.

[13] John McAvoy and Tom Butler "The impact of the Abilene Paradox on double-loop learning in an agile team" Information and Software Technology, Vol. 49, Issue 6, June 2007, pp 552-563.

[14] Ivana Turnu., Marco Melis., and Katiuscia Mannaro "Modeling and simulation of open source development using an agile practice" Journal of Systems Architecture, Vol. 52, Issue 11, Nov 2006, pp 610-618.
[15] Witold Pedrycz "Quantitative logic-based framework for agile methodologies" Journal of Systems Architecture, Vol. 52, Issue 11, November 2006, pp 700-707.

G.Lakshmi Mounica Kakinada Institute of Engineering & Technology Womens, korangi. Presently she is pursuing her M.Tech [Computer Science of Engineering] from this college and he received her B.Tech from chaitanya Institute of Engineering & Technology, affiliated to JNT University, Kakinada in the year 2013. Her area of interest includes Cloud Computing and Object oriented Programming languages,operating systems, all current trends and techniques in Computer Science

T.Sai Durga , well known Author and excellent teacher Received .Tech (CSE) from JNT university KAKINADA is working as Associate Professor in Department of Computer science engineering, Kakinada institute of Engineering and Technology for women. She is an active member of ISTE. She has 8 years of teaching experience in engineering. To her credit couple of publications both national and international conferences /journals . Her area of Interest includes Data Warehouse and Data Mining, computer networks, information security, mobile computing and other advances in computer Applications.